Snow Scene Segmentation Using CNN-Based Approach With Transfer Learning

Parintorn Pooyoi*, Punyanuch Borwarnginn*, Jason H. Haga† and Worapan Kusakunniran*
* Faculty of Information and Communication Technology
Mahidol University, Nakhon Pathom, Thailand
† Information Technology Research Institute
National Institute of Advanced Industrial Science and Technology, Tsukuba, Ibaraki, Japan
Emails: parintorn.poo@gmail.com, punyanuch.bor@mahidol.edu, jh.haga@aist.go.jp, worapan.kun@mahidol.edu

***Abstract*—Images from CCTV cameras can be used for analyzing disaster situations in a particular area. Snowfall is one of the weather conditions that could cause natural disasters in Japan. It is possible for a machine to detect snow and mark these areas in that image. There are existing convolutional neural network-based (CNN-based) frameworks that can achieve high accuracy in an object classification task. However, these frameworks cannot define or mark the affected area then display the result. To address this problem, this paper proposes a method to develop a model using CNN frameworks, with the transfer learning technique. We use transfer learning to reduce training time and computing resources while maintaining high performance in a snow detection task. For transfer learning, the pre-trained weights from the VGG19 dataset is used. In this work, we use images from CCTV cameras, which were obtained from a publicly accessible website in Japan. After evaluating the model we performed post-processing on the data to further reduce the error in the results. Our proposed method can achieve an average sensitivity of 80.90% and specificity 98.54% for snow detection in real-world images.**

***Index Terms*—Snow detection, Deep learning, CNN, Transfer learning**

I. Introduction

Natural disasters are an important global problem affecting many different countries. In Japan, a public website was made available to provide a variety of data from different sensors throughout the country, as shown in Fig. 1. This data includes information about river water levels rainfall levels, and snowfall levels. Furthermore, this information includes CCTV cameras positioned along the river, which provide photos of the river conditions in real-time [1]. These photos report to users a visual snapshot on the current status of the river, but does not provide any additional information or annotation. Therefore, it relies on the users to make their own interpretation and judgment regarding the status in the picture.. Thus, in an effort to provide a better assessment of the status in the images, the goal of this project is to improve the usability of this CCTV image data by using image processing with machine learning.

The first existing related project was introduced by Bossu et al. [2]. The project used a system based on computer vision that detects the presence of snow or rain. It uses a classical Gaussian mixture model to separate the foreground and background image sequences. The authors also proposed a method to use video cameras to detect rain. The classical MoG model was used to separate those images from the background in the image sequences. This project represents work on the segmentation of snowfall in the sky and raindrops that can be compared frame by frame from the provided video capture.

Cloud detection from RGB color remote sensing images with deep pyramid networks [3], is another related project. In the paper, this approach used the deep pyramid network to perform cloud detection from the RGB images. The visual data provided gave a method to generate pixel level decisions using spatial texture information. Moreover, they also examined the integration of the pre-trained CNN model that provided good accuracy of the classification mask at the encoder layer. From the experimental results, these methods quantitatively outperform the baselines and obtains perceptually superior results on the datasets.

These projects do not focus specifically on disaster management. However, they provide more generally some information regarding image processing to detect weather conditions. These methods could be improved to detect and define areas of snow in images.

The rest of this paper is organized as follows. The background knowledge and the proposed method are described in the sections II and III respectively. The results are shown in the section IV and the conclusion is drawn in the section IV.

II. Background knowledge

A. Convolutional neural network

In deep learning, a convolutional neural network (CNN) is a type of deep neural networks [4]. Most CNNs are applied to analyzing visual images. CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing. They use relatively few pre-processing methods compared to other image classification algorithms. This means that the network can learn the filters that are hand-engineered in traditional algorithms. They have been applied to a wide variety of tasks including image and video recognition, recommender systems, image classification, medical image analysis, and natural language processing

978-1-7281-3361-4/19/$31.00 ©2019 IEEE

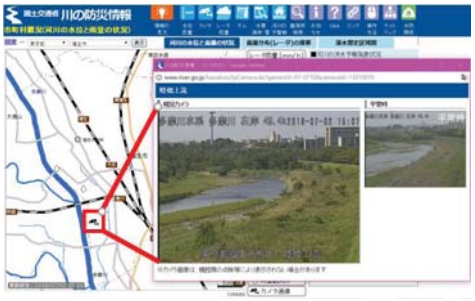

Fig. 1. Screen capture of the Japan website that presents the data to the public.

B. Transfer Learning

Transfer learning [5] [6] is a technique to help reuse the pretrained model that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. This enables the reuse of neural networks for different, but related problems.

III. Proposed Method

The proposed framework for training a model of snow detection is shown in Fig. 2. It is an iterative process that consists of four subprocesses. In the first subprocess, we need to acquire the dataset that can be used for training our detector. The second subprocess is data preparation. Since we want to predict snow area in the image, we have to separate the image into sub-images and define the class of each sub-image manually first. The third subprocess is the data modeling. Our CNN detector will be trained using the data from the previous two sub-processes. If the resulting data model is not satisfactory, the model can be recreated again in this sub-process. The last subprocess is post-processing. If the result from the original model is not satisfactory, this process will improve the result. After performing all of the subprocesses, the results can be evaluated using the average sensitivity and specificity of snow detection of the model.. If the model has low performance, we can reiterate through the processes to improve the model.

A. Data Acquisition

For the data acquisition, the images were from CCTV cameras [1]. Each camera has the same size, place and angle of view, but there are differences in time and situation. We manually selected images during daytime and separated images into snow and non-snow containing images in the dataset in order to facilitate the model training.

B. Data Preparation

For this task, we selected 45 images from 3 cameras, 12 snow images and 3 non-snow images per camera in Fig. 3. Before using these data we defined the area by manually converting snow and non-snow area into black and white (B&W) images. Labels were applied to the B&W images defining white areas as snow, and black areas as non-snow.

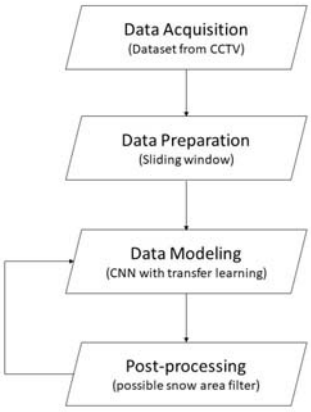

Fig. 2. The proposed framework for training a model of snow detection.

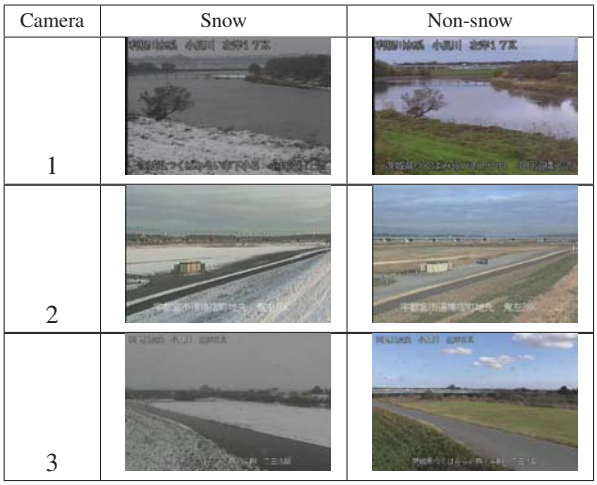

Fig. 3. Examples from the dataset depicting images that had snow and had no-snow.

With both the original and converted images, we then used a "sliding window" [7] [8] to create a series of small (50 x 50 pixel area) sub-images in the original image and B&W image as shown in Fig. 4. Then every pixel in the labeled B&W sub-image is checked for snow with the following criteria: if more than 60% of the pixels are white, the sub-image is classified as a snow area, otherwise it will classify it as non-snow. However, there is a range of percent pixels between 40-60% that causes ambiguity for the modeling and is not used. After classifying all the sub-images, the original image will be cropped into corresponding boxes and saved into new folders as either snow or non-snow. This process is repeated for the entire original image by sliding the window to the right in this case, which overlaps slightly with the previous window position and then classifying each sub-image as snow or non-snow using the 60% white pixel criteria.

C. Data Modeling

To create the model for this work, we selected a convolutional neural network [9] as illustrated in Fig. 5. There are sev-

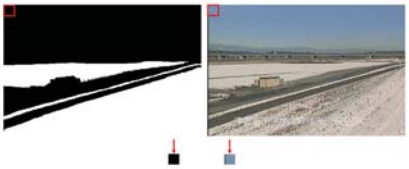

Fig. 4. Sliding window method to classify areas of the image as snow or non-snow.

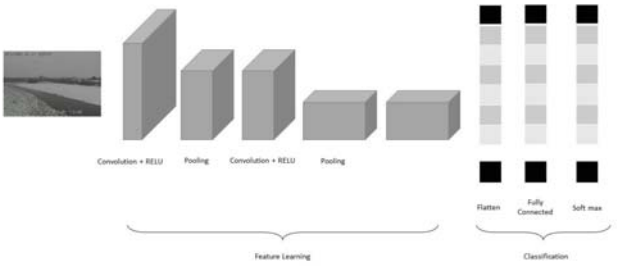

Fig. 5. Architecture of CNN.

eral existing CNN architectures but VGG-19 [10] [11] is used in our proposed framework because the architecture is suitable for the image classification task. VGG-19 is widely used in image classification because of its general performance. From the sliding window, we generated and used 45,000 images per each class for training. We loaded the VGG-19 pre-train weights model then let the model calculate new weights from the training data. The output from this process is a model that can classify an input image into predefined classes during the training procedure.

D. Post-processing

Applying this model to an image, generates a result that shows black and white labeled snow areas. The results can contain two types of errors: a false positive (model detects a non-snow area as snow) and false negative (model detects a snow area as non-snow). To reduce the error we created a post-processing filter called possible snow area to remove false positive. This filter is performed manually based on human experience and replaces white pixel with black pixel in areas that cannot have snow such as in the sky or in the water. This process will increase the overall accuracy of the model. Fig. 6 shows an example of false positive image where the clouds in the sky are detected as snow. The post-processing process is shown in Fig. 7.

IV. Results

The test dataset used in this project are images from the same three CCTV cameras in Japan. It contains 20 images per camera, 10 for snow and 10 for non-snow. All of the test data images are generated from the sliding window and are not used for training the model. In this work we compare two sets of results the original result and post-processed

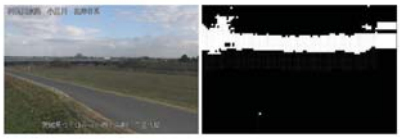

Fig. 6. Example of false positive result where white clouds create white pixels that are misclassified as snow. (white area at top of image on right)

Fig. 7. Process of post-processing start from original image, original label by model, human filter, and after post-processing

results. The metrics used to evaluate the performance are sensitivity and specificity. We compared the results between human classification and predicted classification yielding four types of outcomes as follows:

- True positive is the number of pixels that detect snow area as snow.
- True negative is the number of pixels that detect non-snow area as non-snow.
- False positive is the number of pixels that detect non-snow area as snow.
- False negative is the number of pixels that detect snow area as non-snow.

These results are counted by comparing every pixel in the image before calculating the sensitivity and specificity. The sensitivity shows the true positive percentage and is calculated by this equation:

$$\frac{TP}{TP + FN} \tag{1}$$

where TP = true positive, and FN = false negative.

The specificity shows the true negative percentage and is calculated by this equation:

$$\frac{TN}{TN + FP} \tag{2}$$

where TN = true negative, and FP = false positive.

Both sensitivity and specificity can be a value between 0 and 1 where 1 is the highest result or 100%.

A. Results with pre-trained weights

The first experiment is to model the data using pre-trained weights of the VGG-19 dataset then apply this model to our test dataset. From 60 images, we can achieve an average sensitivity of 78.30% and a specificity of 78.36%. These results show that the sensitivity is low, or that the false positive percentage (i.e. detect non-snow area as snow) is high as shown in Table I.

B. Results with pre-trained weight and apply post-processing

In this experimental scenario, we applied the post-processing by using a filter to decrease the false positives from the result of the first experiment. From 60 images, we

TABLE I
EXPERIMENTAL RESULTS REPORTED USING THE EXAMPLES FROM THE DATASET DEPICTING IMAGES THAT HAD SNOW AND HAD NO-SNOW.

	TP	TN	FP	FN	Sensitivity	Specificity
Scenario1(a): Images containing snow without the post-processing (average±SD)	27.42±4.49%	35.36±5.98%	29.03±7.56%	8.17±6.97%	78.30±0.17%	55.17±0.08%
Scenario1(b): Images containing non snow without the post-processing (average±SD)	n/a	93.46±8.88%	6.53±8.88%	n/a	n/a	93.46±0.06%
Scenario2(a): Images containing snow with the post-processing (average±SD)	29.28±3.42%	61.42±4.92%	1.89±1.28%	7.39±5.88%	80.90±0.13%	96.88±0.02%
Scenario2(b): Images containing non snow with the post-processing (average±SD)	n/a	98.20±2.79%	1.79±2.79%	n/a	n/a	98.20±0.02%
Average±SD of scenario 1 (without post-processing)	13.71±14.18%	64.51±30.23%	17.78±13.98%	4.09±6.39%	78.30±0.17%	78.36±0.05%
Average±SD of scenario 2 (with post-processing)	14.64±14.95%	79.81±18.96%	1.84±2.15%	3.69±5.56%	80.90±0.13%	98.54±0.01%

achieved an average sensitivity of 80.90% and specificity of 98.54% respectively as shown in TABLE I. These results show that the post-processing improves the results when compared to using the model alone.

The snow detection can be trained by retraining the network with the pre-trained weight of VGG19. However, the trained model is impacted by the occurrence of false positives (detect non-snow as snow). This makes the model unable to adequately define the true area in the image without the aid of a post-processing filter. On the other hand, for true negatives (detect non-snow area as non-snow) the model performed with an with average specificity of 93.46% using the model alone without a post-processing filter. This result suggests that it is easier to define non-snow areas with the training data versus snow areas. Another factor that might affect the results is the number of images in the dataset. As mentioned in the Data Preparation section, only 45 images were used to generate the dataset using the sliding window to create sub-images. The final number of images in the training dataset was approximately 45,000 sub-images. The result might be improved if we increase the number of data sub-images, but this could also lead to overfitting the model.

V. CONCLUSION

This paper proposed a snow detection method by using CNN and transfer learning techniques. The proposed approach consists of a process of data acquisition, data preparation, data modeling, post-processing, and evaluation. Using publicly available images from CCTV cameras in Japan as a dataset, Training and test datasets were prepared using a sliding window. A CNN is used to model snow and non-snow areas with the VGG19 pre-trained weights We achieved a model with the best average sensitivity of 78.30% and specificity of 78.36%. After that we implemented a post-processing to reduce the number of false positive and the average sensitivity improved to 80.90% and the average specificity improved to 98.54%. In the future, more variety of images of snow and non-snow from another CCTV cameras will be incorporated into the training dataset to create a new model that could be more robust to false positive. This will result in a model that will work with more varied situations in the images and improve the classification ability of this model for detecting snow.

ACKNOWLEDGMENT

This research project was partially supported by Faculty of Information and Communication Technology, Mahidol University, and the ICT International Collaboration Fund of the National Institute of Advanced Industrial Science and Technology.

REFERENCES

[1] "Disaster prevention information of river," http://www.river.go.jp/kawabou/ipTopGaikyo.do, accessed on 2019-02-03.
[2] J. Bossu, N. Hautière, and J.-P. Tarel, "Rain or snow detection in image sequences through use of a histogram of orientation of streaks," *International Journal of Computer Vision*, vol. 93, no. 3, pp. 348–367, 2011.
[3] S. Ozkan, M. Efendioglu, and C. Demirpolat, "Cloud detection from rgb color remote sensing images with deep pyramid networks," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 6939–6942.
[4] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.
[5] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, 2014, pp. 3320–3328.
[6] S. Thrun and L. Pratt, "Transfer learning," https://en.wikipedia.org/wiki/Transfer_learning, December 2012, accessed on 2019-02-03.
[7] A. Rosebrock, "Sliding window," https://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/, March 2015, accessed on 2019-02-03.
[8] C. Wojek, G. Dorkó, A. Schulz, and B. Schiele, "Sliding-windows for rapid object class localization: A parallel technique," in *Joint Pattern Recognition Symposium*. Springer, 2008, pp. 71–81.
[9] P. Raghav, "Understanding of convolutional neural network (cnn)-deep learning," https://medium.com/@RaghavPrabhu, March 2018, accessed on 2019-02-03.
[10] "Keras documentation vgg19 original source," https://keras.io/applications/#vgg19, accessed on 2019-02-03.
[11] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.